(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) N° de publication : **2 830 146**
(à n'utiliser que pour les commandes de reproduction)

(21) N° d'enregistrement national : **01 12274**

(51) Int Cl[7] : H 04 L 9/06, G 06 K 19/07

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt** : 24.09.01.

(30) **Priorité** :

(43) **Date de mise à la disposition du public de la demande** : 28.03.03 Bulletin 03/13.

(56) **Liste des documents cités dans le rapport de recherche préliminaire** : *Se reporter à la fin du présent fascicule*

(60) **Références à d'autres documents nationaux apparentés** :

(71) **Demandeur(s)** : *GEMPLUS Société anonyme* — FR.

(72) **Inventeur(s)** : CHEVALLIER MAMES BENOIT, FEYT NATHALIE, JOYE MARC, PAILLIER PASCAL, QUES FLORENCE et GANDOLFI KARINE ep. VILLEGAS.

(73) **Titulaire(s)** :

(74) **Mandataire(s)** :

(54) PROCEDE DE MISE EN OEUVRE, DANS UN COMPOSANT ELECTRONIQUE, D'UN ALGORITHME DE CRYPTOGRAPHIE ET COMPOSANT CORRESPONDANT.

(57) L'invention concerne un procédé de mise en oeuvre, dans un composant électronique, d'un algorithme de cryptographie utilisant des moyens de calcul, caractérisé en ce que qu'il consiste à réaliser les étapes suivantes:

a) choisir une valeur e parmi un nombre déterminé de valeurs e_i, e_i étant des nombres entiers,

b) tester si e_i vérifie une relation prédéterminée:

si c'est le cas, alors e=ei, et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie.

FR 2 830 146 - A1

PROCEDE DE MISE EN ŒUVRE, DANS UN COMPOSANT ELECTRONIQUE, D'UN ALGORITHME DE CRYPTOGRAPHIE ET COMPOSANT CORRESPONDANT

L'invention concerne un procédé de mise en œuvre, dans un composant électronique, d'un algorithme de cryptographie.

L'invention se rapporte également au composant électronique correspondant.

De tels composants sont utilisés dans des applications où l'accès à des services ou à des données est sévèrement contrôlé. Ils ont une architecture formée autour d'un microprocesseur et de mémoires, dont une mémoire programme de type ROM ("Read Only Memory" en anglais) qui contient le(s) nombre(s) secret(s) d.

Ces composants sont utilisés dans des systèmes informatiques, embarqués ou non ; ils sont notamment utilisés dans les cartes à puce, pour certaines applications de celles-ci. Ce sont par exemple des applications d'accès à certaines banques de données, des applications bancaires, des applications de télépéage, par exemple pour la télévision, la distribution d'essence ou encore le passage de péages d'autoroutes.

Ces composants ou ces cartes mettent donc en œuvre un algorithme de cryptographie pour assurer le chiffrement de données émises et/ou le déchiffrement de données reçues lorsque celles-ci doivent demeurer confidentielles.

De manière générale et succincte, ces algorithmes cryptographiques ont notamment pour fonction le

chiffrement ou la signature numérique d'un message. A partir de ce message appliqué en entrée à la carte par un système hôte (serveur, distributeur bancaire...) et de nombreux secrets contenus dans la carte, la carte fournit en retour au système hôte le message chiffré ou signé, ce qui permet par exemple au système hôte d'authentifier le composant ou la carte, d'échanger des données, ...

Les caractéristiques des algorithmes de cryptographie sont connues : calculs effectués, paramètres utilisés. La seule inconnue est le ou les nombres secrets contenus en mémoire programme. Toute la sécurité de ces algorithmes de cryptographie tient dans ce(s) nombre(s) secret(s) contenu(s) dans la carte et inconnu(s) du monde extérieur à cette carte. Ce nombre secret ne peut être déduit de la seule connaissance du message appliqué en entrée et du message chiffré fourni en retour.

Or il est apparu que des attaques externes permettent à des tiers mal intentionnés de trouver le(s) nombre(s) secret(s) contenu(s) dans cette carte. Dans le domaine de la carte à puce, entre autres, il existe plusieurs attaques possibles, dont une dite "attaque par faute".

Dans ce type d'attaque, l'attaquant injecte une faute quelconque pendant le calcul d'un algorithme cryptographique, dans le but d'exploiter la présence de cette faute pour extraire une information secrète.

La faute peut aussi provenir d'une erreur de calcul due au matériel mettant en œuvre l'algorithme cryptographique ; on considère néanmoins, dans un cas

comme dans l'autre, qu'il s'agit d'une attaque par faute.

Ce type d'attaque est notamment envisageable avec l'algorithme RSA (du nom de ses auteurs Rivert, Shamir, Adleman), qui est celui le plus utilisé en cryptographie dans ce domaine d'application. La sécurité de l'algorithme RSA est basée sur la difficulté de factoriser de grands nombres. Ces algorithmes utilisent notamment des calculs d'exponentiation à la puissance d, d étant un nombre secret.

On rappelle brièvement les principales étapes de l'algorithme RSA.

On établit un nombre N qui est le produit de deux nombres premiers p et q (N=p.q), ainsi qu'un exposant public ou clé publique e et un exposant privé ou clé privée ou secrète d, satisfaisant la relation :

$$e.d = 1 \text{ (modulo } \lambda(N)), \qquad (1)$$

$\lambda(.)$ étant la fonction de Carmichael.

Selon un premier mode de fonctionnement de l'algorithme RSA dit standard, les paramètres publics sont (N,e) et les paramètres privés sont (N,d). Etant donné x compris dans l'intervalle]0,N[, l'opération publique sur x qui peut être par exemple le chiffrement du message x ou encore la vérification de la signature x, consiste à calculer :

$$y = x^e \text{ modulo } N \qquad (2)$$

L'opération privée correspondante qui peut être par exemple le déchiffrement du message chiffré y ou la génération d'une signature x, consiste à calculer :

y^d modulo N (3)

avec $x = y^d$ modulo N puisque e.d = 1(modulo $\lambda(N)$).

On va présenter un autre mode de fonctionnement dit mode CRT car basé sur le théorème des restes chinois ("Chinese Remainder Theorem" ou CRT en anglais) et quatre fois plus rapide que celui de l'algorithme RSA standard. Selon ce RSA mode CRT, on n'effectue pas directement les calculs modulo N mais on effectue d'abord les calculs modulo p et modulo q.

Les paramètres publics sont (N,e) mais les paramètres privés sont (p,q,d) ou (p,q,d_p,d_q,i_q) avec

$d_p = d$ modulo$(p-1)$, $d_q = d$ modulo$(q-1)$

et $i_q = q^{-1}$ modulo p.

Par la relation (1), on obtient :

$ed_p=1$ modulo$(p-1)$ et $ed_q=1$ modulo$(q-1)$ (4)

L'opération publique s'effectue de la même façon que pour le mode de fonctionnement standard ; par contre pour l'opération privée, on calcule d'abord :

$$x_p = y^{d_p} \text{mod} p \quad \text{et} \quad x_q = y^{d_q} \text{mod} q$$

Ensuite, par application du théorème des restes chinois, on obtient $x = y^d$ mod N par :

$$x = CRT(x_p, x_q) = x_q + q[i_q(x_p - x_q) \text{ modulo } p] \quad (5)$$

L'algorithme RSA a été présenté avec deux facteurs premiers p et q, pour simplifier l'exposé. On peut le généraliser au cas où N est le produit de deux entiers p et q tels que pgcd(p,q)=1. Dans ce cas,

$d_p = d$ (modulo $\lambda(p)$), $d_q = d$ (modulo $\lambda(q)$),

i_q reste inchangé par rapport au cas précédent,

$ed_p=1$ (modulo $\lambda(p)$) et $ed_q=1$ (modulo $\lambda(q)$),

et les calculs de x_p, x_q et x sont inchangés.

Cette généralisation s'applique au mode standard comme au mode CRT.

On va à présent décrire un exemple d'attaque par faute basée sur l'obtention de deux signatures du même message, l'une correcte x et l'autre incorrecte notée ^x.

La signature incorrecte a par exemple été obtenue de la manière suivante. L'attaquant, par une méthode quelconque, injecte une erreur durant le calcul de x_p, mais pas durant celui de x_q. La valeur de x_p est alors incorrecte et notée $\hat{x}_p$. Par contre, la valeur de x_q est correcte. De ce fait, lorsque l'on recombine les valeurs $\hat{x}_p$ et x_q en appliquant le théorème des restes chinois, la signature résultante ^x est incorrecte.

Il suffit alors à l'attaquant qui connaît bien sûr par ailleurs les paramètres publics (N,e), de calculer le plus grand commun diviseur (pgcd) avec N, soit :

pgdc(^x - x, N).

Or, pgdc(^x - x, N)= q. Il obtient alors le facteur secret q et donc p et d_p et d_q. De ce fait, le code RSA est effectivement cassé.

Autrement dit, si quelqu'un est capable d'injecter une erreur quelconque durant un calcul modulo p alors que le calcul modulo q est correct ou réciproquement, il peut casser complètement le code RSA.

Il est également possible de casser le code RSA à partir d'une signature incorrecte d'un message connu. Plusieurs cas d'attaques par faute sont présentés dans la publication "On the Importance of Checking Cryptographic Protocols for Faults" de D. Boneh, R.A.

DeMillo et R.J. Lipton , Advances in Cryptology-EUROCRYPT'97 pp.37-51, à laquelle on peut se reporter.

Une première contremesure pour éviter ce genre de scénario consiste à recalculer l'ensemble de l'algorithme. On compare les valeurs obtenues à l'issue des calculs successifs. S'ils sont identiques, on suppose qu'il n'y a pas eu d'erreur injectée. Un problème avec cette approche est qu'elle ne détecte pas une faute permanente. Par exemple, on ne peut déceler une attaque dans laquelle l'erreur injectée consiste à ce que la valeur d'un bit de mémoire soit toujours fixée à 0 ou à 1 ("sticky bit" en anglais).

Une autre contre-mesure à l'attaque par faute est décrite par Shamir dans le document brevet WO 98 52319.

On procède selon cette contre-mesure par l'algorithme suivant :

1. Choisir un nombre aléatoire r de faible valeur,
2. Calculer :

$x_{rp} = y^d$ modulo r.p, et

$x_{rq} = y^d$ modulo r.q ;

3. Si $x_{rp} \neq x_{rq}$ (modulo r), alors il y a erreur, (peut-être induite par une attaque,) et donc interruption de l'algorithme, sinon ;
4. Appliquer le théorème des restes chinois à x_{rp} et x_{rq}, pour émettre x en sortie.

On effectue ainsi respectivement les calculs modulo r.p et modulo r.q au lieu de modulo p et modulo q. Ensuite, on vérifie que les deux valeurs x_{rp} et x_{rq} obtenues par ces calculs sont égales modulo r. Si ces deux valeurs sont différentes, il est certain qu'il y a

eu une erreur. Par contre, si elles sont égales, on peut supposer qu'il n'y a pas eu d'erreur, avec une probabilité de 1/r de se tromper dans cette supposition.

Un inconvénient de cette méthode est qu'elle est probabiliste, c'est-à-dire que les erreurs sont détectées avec une probabilité inférieure à 1 et que par conséquent, elles ne sont pas toutes détectées. En outre, elle est coûteuse en temps de calcul. Un autre inconvénient de la méthode Shamir est qu'elle ne fonctionne que pour le mode CRT. Or, il est également envisageable d'utiliser le mode standard de l'algorithme RSA.

La meilleure protection possible pour se protéger des attaques par faute consiste à vérifier que la valeur x obtenue lors de l'opération privée (3) ou (5) (c'est-à-dire soit en mode standard, soit en mode CRT de l'algorithme RSA), vérifie la relation (2) $y = x^e$ modulo N de l'opération publique. En effet, lorsque cette relation est vérifiée, on est assuré qu'il n'y a pas eu d'erreur pendant le déroulement de l'opération privée de l'algorithme RSA.

Mais le composant ou le dispositif mettant en œuvre l'opération privée ne dispose pas toujours de l'exposant public e, notamment lorsqu'il n'exécute que l'opération privée.

Au vu de ce qui précède, l'invention propose un procédé permettant de réaliser certaines étapes d'un algorithme de cryptographie utilisant un exposant public e que l'on ne connaît pas a priori.

Ce procédé permet en particulier de réaliser une contre-mesure, notamment aux attaques par fautes, qui offre la meilleure protection possible même lorsqu'on ne connaît pas l'exposant public e.

L'invention a pour objet un procédé de mise en œuvre, dans un composant électronique, d'un algorithme de cryptographie utilisant des moyens de calcul, principalement caractérisé en ce que qu'il consiste à réaliser les étapes suivantes :

a) choisir une valeur e parmi un nombre déterminé de valeurs e_i, e_i étant des nombres entiers,

b) tester si la valeur e_i choisie vérifie une relation prédéterminée:

- si c'est le cas, alors $e=e_i$, et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

- si ce n'est pas le cas, réitérer les étapes précédentes en choisissant une autre valeur de e_i et si aucune valeur de e_i ne peut être attribuée à e alors constater que les calculs dudit algorithme de cryptographie utilisant la valeur e ne peuvent être effectués.

Selon un mode de réalisation de l'invention, il consiste préalablement à l'étape b), à choisir une valeur Y comprise dans l'intervalle]0,N[et à attribuer à une valeur X le résultat de l'opération Y^d modulo N, d et N étant des nombres entiers déterminés et il consiste en ce que la relation prédéterminée de l'étape b) est

$$X^{e_i} \bmod N = Y$$

On choisit de préférence Y=2.

L'algorithme de cryptographie peut être basé sur un algorithme de type RSA, notamment en mode standard ou en mode CRT.

Selon un autre mode de réalisation, la relation prédéterminée de l'étape b) est : $e_i d_p = 1$ (modulo $\lambda(p)$), p et d_p étant des nombres entiers déterminés et $\lambda(.)$ étant la fonction de Carmichael.

Le nombre d_p peut être obtenu par $d_p = d$ (modulo $\lambda(p)$), d étant un entier prédéterminé.

Selon une caractéristique de l'invention, d_q et q étant des nombres entiers déterminés, avec pgcd(p,q)=1, l'étape b) consiste à réaliser les étapes suivantes :

tester si $e_i d_p = 1$ (modulo $\lambda(p)$),

si c'est le cas, et si $e_i < \lambda(p)$, alors $e = e_i$ et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

si c'est le cas, et si $e_i \geq \lambda(p)$, alors tester si $e_i d_q = 1$ (modulo $\lambda(q)$); si c'est le cas, alors $e = e_i$ et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

si l'un des deux tests précédents n'est pas vérifié, réitérer les étapes précédentes en choisissant une autre valeur de e_i et si aucune valeur de e_i ne peut être attribuée à e alors constater que les calculs dudit algorithme de cryptographie utilisant la valeur e ne peuvent être effectués.

Le nombre d_q peut être obtenu par $d_q = d$ (modulo $\lambda(q)$), d étant un entier prédéterminé.

L'algorithme de cryptographie est avantageusement basé sur un algorithme de type RSA en mode CRT.

On choisit de préférence $e_i = 2^{16}+1$ ou $e_i=3$.

Selon une caractéristique de l'invention, une valeur e_i ayant été attribuée à e, il consiste à obtenir à l'issue d'une opération privée de l'algorithme RSA, une valeur x à partir d'une valeur y et à ce que ledit calcul utilisant la valeur e consiste à vérifier si $y = x^e$ modulo N, N étant un nombre entier prédéterminé.

L'invention a aussi pour objet un composant électronique de sécurité, comprenant des moyens de calcul, une mémoire de programme et une mémoire de travail et des moyens de communication de données, caractérisé en ce qu'il met en œuvre le procédé tel que précédemment décrit.

L'invention se rapporte en particulier à une carte à puce comprenant un composant électronique tel que décrit ci-dessus.

D'autres particularités et avantages de l'invention apparaîtront clairement à la lecture de la description faite à titre d'exemple non limitatif et en regard de la figure 1 annexée qui représente schématiquement les éléments d'une carte à puce apte à mettre en œuvre l'invention.

Les modes de réalisation sont décrits dans le cadre de cartes à puce, mais peuvent bien entendu s'appliquer à tout autre dispositif ou composant électronique de sécurité doté de moyens de calculs cryptographiques.

Ainsi que le montre la figure 1, la carte à puce 1 comprend un microprocesseur 2 couplé à une mémoire

figée (ROM) 3 et à une mémoire vive (RAM) 4, le tout formant un ensemble permettant, entre autres, l'exécution d'algorithmes cryptographiques. Plus précisément, le microprocesseur 2 comporte les moyens de calcul arithmétiques nécessaires à l'algorithme, ainsi que des circuits de transfert de données avec les mémoires 3 et 4. La mémoire figée 3 contient le programme exécutoire de l'algorithme cryptographique sous forme de code source, alors que la mémoire vive 4 comporte des registres pouvant être mis à jour pour le stockage de résultats des calculs.

La carte à puce 1 comporte aussi une interface de communication 5 reliée au microprocesseur 2 pour permettre l'échange de données avec l'environnement extérieur. L'interface de communication 5 peut être du type "à contacts", étant dans ce cas formée d'un ensemble de plots de contacts destinés à se connecter à un contacteur d'un dispositif externe, tel qu'un lecteur de cartes, et/ou du type "sans contact". Dans ce dernier cas, l'interface de communication 5 comporte une antenne et des circuits de communication par voie hertzienne permettant un transfert de données par liaison sans fil. Cette liaison peut aussi permettre un transfert d'énergie d'alimentation des circuits de la carte 1.

On va à présent décrire une méthode permettant de valider la valeur d'un exposant public e que l'on ne connaît pas a priori.

Elle est basée sur la constatation suivante : dans 90% des cas, la valeur de e est $e_0=2^{16}+1$, dans 5% des

cas, la valeur de e est $e_1=3$ et dans les autres cas, la valeur de e est autre.

La méthode consiste alors à choisir e_0 et à vérifier que $e=e_0$; si $e \neq e_0$, alors on essaie avec e_1.

Il se peut que pour une certaine application correspondant aux 5% d'autres cas, e ne soit pas égal à e_0 ni à e_1. Aussi désigne-t-on plus généralement la valeur de e par e_i. Et la méthode consiste finalement à choisir une valeur e_i parmi les e_i envisagés et à vérifier que $e=e_i$.

Selon un premier mode de réalisation, valable pour les modes standard ou CRT de l'algorithme RSA :

on choisit arbitrairement une valeur Y comprise dans l'intervalle]0,N[,

on choisit une valeur e_i,

on calcule $X = Y^d$ modulo N par (3) en mode standard ou par (5) en mode CRT

$$\text{si } X^{e_i} \bmod N = Y \text{ alors } e = e_i$$

et on mémorise e

sinon on choisit une autre valeur pour e_i.

On peut avantageusement choisir Y=2 de manière à accélérer le calcul d'exponentiation Y^d qui apparaît dans la relation (3) ou (5) : cela revient alors à faire des additions au lieu de multiplications.

On décrit à présent un autre mode de réalisation basé sur la relation (4) ; il n'est valable qu'en mode CRT mais est alors plus efficace que le mode de réalisation précédent :

on choisit une valeur e_i,

on teste si $e_i d_p=1$ modulo(p-1), (ou si $e_i d_p=1$ (modulo $\lambda(p)$) dans le cas général)

si oui et si $e_i<p$ (ou si $e_i<\lambda(p)$ dans le cas général), alors $e=e_i$ et on mémorise e

si oui et si $e_i \geq p$, (ou si $e_i \geq \lambda(p)$ dans le cas général) alors $e=e_i$ avec une très grande probabilité de l'ordre de 1-2/p.

Dans le cas où $e_i \geq p$ (ou si $e_i \geq \lambda(p)$ dans le cas général), l'ambiguïté peut être levée avec une probabilité égale à 1 en testant si $e_i d_q=1$ modulo(q-1)(ou si $e_i d_q=1$ (modulo $\lambda(q)$)dans le cas général). Si c'est le cas, $e=e_i$ et on mémorise e.

Cependant, dans la majorité des cas ($e_i=2^{16}+1$ ou $e_i=3$), $e_i<p$ (ou $e_i<\lambda(p)$ dans le cas général) car p a une taille de 512 bits ou plus.

Si l'un des tests n'est pas vérifié, on choisit une autre valeur pour e_i.

Si pour l'un ou l'autre mode de réalisation, il n'existe pas parmi les e_i, une valeur telle que $e=e_i$, alors on ne peut effectuer les calculs faisant intervenir e.

Lorsqu'on connaît e, par l'un ou l'autre de ces modes de réalisation, on peut alors vérifier chaque opération privée (3) ou (5) en s'assurant que $y=x^e$ modulo N ou plus généralement effectuer des calculs utilisant la valeur e qui est mémorisée.

Comme on l'a vu, ce procédé peut bien sûr être appliqué à une contre-mesure.

Il est plus rapide que la contre-mesure décrite dans l'état de la technique et qui consiste à

recalculer l'ensemble de l'algorithme, c'est-à-dire à d'effectuer au moins un deuxième calcul d'exponentiation à la puissance d, d étant de la taille de N, et à comparer les valeurs obtenues à l'issue des calculs successifs. Le procédé selon l'invention consiste aussi à effectuer un deuxième calcul d'exponentiation mais à la puissance e ; or e est petit.

Il permet en outre de détecter une faute permanente.

Il s'applique aussi bien dans le cas du mode standard de l'algorithme RSA que dans le cas du mode CRT ainsi qu'à la généralisation de ces modes.

REVENDICATIONS

1. Procédé de mise en œuvre, dans un composant électronique, d'un algorithme de cryptographie utilisant des moyens de calcul, caractérisé en ce que qu'il consiste à réaliser les étapes suivantes :

a) choisir une valeur e parmi un nombre déterminé de valeurs e_i, e_i étant des nombres entiers,

b) tester si la valeur e_i choisie vérifie une relation prédéterminée:

- si c'est le cas, alors $e=e_i$, et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

- si ce n'est pas le cas, réitérer les étapes précédentes en choisissant une autre valeur de e_i et si aucune valeur de e_i ne peut être attribuée à e alors constater que les calculs dudit algorithme de cryptographie utilisant la valeur e ne peuvent être effectués.

2. Procédé selon la revendication précédente, caractérisé en ce que préalablement à l'étape b), il consiste à choisir une valeur Y comprise dans l'intervalle]0,N[et à attribuer à une valeur X le résultat de l'opération Y^d modulo N, d et N étant des nombres entiers déterminés et en ce que la relation prédéterminée de l'étape b) est

$$X^{e_i} \bmod N = Y$$

3. Procédé selon la revendication précédente, caractérisé en ce que Y=2.

4. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que l'algorithme de cryptographie est basé sur un algorithme de type RSA.

5. Procédé selon la revendication précédente, caractérisé en ce que l'algorithme de type RSA est en mode standard ou en mode CRT.

6. Procédé selon la revendication 1, caractérisé en ce que la relation prédéterminée de l'étape b) est : $e_i d_p=1$ (modulo $\lambda(p)$), p et d_p étant des nombres entiers déterminés et $\lambda(.)$ étant la fonction de Carmichael.

7. Procédé selon la revendication précédente, caractérisé en ce que $d_p=d$ (modulo $\lambda(p)$), d étant un entier prédéterminé.

8. Procédé selon l'une quelconque des revendications 6 ou 7, caractérisé en ce que, d_q et q étant des nombres entiers déterminés, avec pgcd(p,q)=1, l'étape b) consiste à réaliser les étapes suivantes :

tester si $e_i d_p=1$ (modulo $\lambda(p)$),

si c'est le cas, et si $e_i<\lambda(p)$, alors $e=e_i$ et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

si c'est le cas, et si $e_i \geq \lambda(p)$, alors tester si $e_i d_q = 1$ (modulo $\lambda(q)$); si c'est le cas, alors $e = e_i$ et mémoriser e en vue de son utilisation dans des calculs dudit algorithme de cryptographie,

si l'un des deux tests précédents n'est pas vérifié, réitérer les étapes précédentes en choisissant une autre valeur de e_i et si aucune valeur de e_i ne peut être attribuée à e alors constater que les calculs dudit algorithme de cryptographie utilisant la valeur e ne peuvent être effectués.

9. Procédé selon la revendication précédente caractérisé en ce que $d_q = d$ (modulo $\lambda(q)$), d étant un entier prédéterminé.

10. Procédé selon l'une quelconque des revendications 6 à 9, caractérisé en ce que l'algorithme de cryptographie est basé sur un algorithme de type RSA en mode CRT.

11. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que

$$e_i = 2^{16}+1.$$

12. Procédé selon l'une quelconque des revendications 1 à 10, caractérisé en ce que $e_i = 3$.

13. Procédé selon l'une quelconque des revendications précédentes et selon lequel une valeur e_i a été attribuée à e, caractérisé en ce qu'il consiste à

obtenir à l'issue d'une opération privée de l'algorithme RSA, une valeur x à partir d'une valeur y et en ce que ledit calcul utilisant la valeur e consiste à vérifier si $y = x^e$ modulo N, N étant un nombre entier prédéterminé.

14. Composant électronique de sécurité, comprenant des moyens de calcul (2), une mémoire de programme (3) et une mémoire de travail (4) et des moyens de communication de données (5), caractérisé en ce qu'il met en œuvre le procédé selon l'une quelconque des revendications précédentes.

15. Carte à puce comprenant un composant électronique selon la revendication précédente.

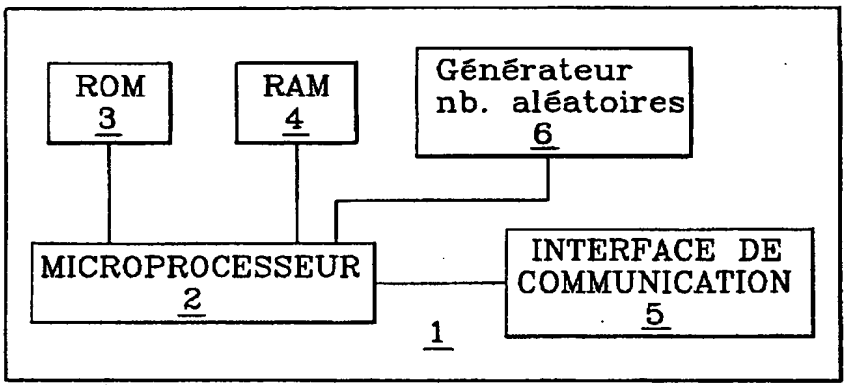
ROM
3
RAM
4
Générateur
nb. aléatoires
6
MICROPROCESSEUR
2
INTERFACE DE
COMMUNICATION
5
1

Fig. 1

RÉPUBLIQUE FRANÇAISE

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

2830146

RAPPORT DE RECHERCHE PRÉLIMINAIRE

établi sur la base des dernières revendications déposées avant le commencement de la recherche

N° d'enregistrement national

FA 610000
FR 0112274

DOCUMENTS CONSIDÉRÉS COMME PERTINENTS		Revendication(s) concernée(s)	Classement attribué à l'invention par l'INPI
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes		
A	EP 0 202 768 A (IBM) 26 novembre 1986 (1986-11-26) * colonne 11, ligne 22 - ligne 41 * * colonne 12, ligne 30 - ligne 40 * * page 8, ligne 6 * * colonne 22, ligne 48 - colonne 23, ligne 16 * ---	1,4-7	H04L9/06 G06K19/07
A,D	BONEH D ET AL: "On the importance of checking cryptographic protocols for faults" ADVANCES IN CRYPTOLOGY - EUROCRYPT '97. INTERNATIONAL CONFERENCE ON THE THEORY AND APPLICATION OF CRYPTOGRAPHIC TECHNIQUES PROCEEDINGS, ADVANCES IN CRYPTOLOGY - EUROCRYPT '97. INTERNATIONAL CONFERENCE ON THE THEORY AND APPLICATION OF CRYPTOGRAPHIC TE, pages 37-51, XP002202745 1997, Berlin, Germany, Springer-Verlag, Germany ISBN: 3-540-62975-0 * page 49, ligne 14 - ligne 21 * ---	2-13	DOMAINES TECHNIQUES RECHERCHÉS (Int.Cl.7) H04L
A	WO 99 35782 A (CRYPTOGRAPHY RESEARCH INC) 15 juillet 1999 (1999-07-15) * page 1, ligne 5 - page 3, ligne 29 * * page 22, ligne 22 - page 24, ligne 12 * ---	2-15	
A,D	WO 98 52319 A (YEDA RES & DEV ;FLEIT LOIS (US)) 19 novembre 1998 (1998-11-19) * page 7, ligne 20 - page 9, ligne 9 * * page 12, ligne 6 - page 13, ligne 21 * -----	2-15	

Date d'achèvement de la recherche	Examinateur
19 juin 2002	Liebhardt, I

CATÉGORIE DES DOCUMENTS CITÉS

X : particulièrement pertinent à lui seul
Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie
A : arrière-plan technologique
O : divulgation non-écrite
P : document intercalaire

T : théorie ou principe à la base de l'invention
E : document de brevet bénéficiant d'une date antérieure à la date de dépôt et qui n'a été publié qu'à cette date de dépôt ou qu'à une date postérieure.
D : cité dans la demande
L : cité pour d'autres raisons

& : membre de la même famille, document correspondant

1

EPO FORM 1503 12.99 (P04C14)

ANNEXE AU RAPPORT DE RECHERCHE PRÉLIMINAIRE RELATIF A LA DEMANDE DE BREVET FRANÇAIS NO. FR 0112274 FA 610000

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche préliminaire visé ci-dessus.
Les dits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du 19-06-2002
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets, ni de l'Administration française

Document brevet cité au rapport de recherche		Date de publication		Membre(s) de la famille de brevet(s)	Date de publication
EP 0202768	A	26-11-1986	DE	3685987 D1	20-08-1992
			DE	3685987 T2	04-02-1993
			EP	0202768 A2	26-11-1986
			US	4736423 A	05-04-1988
WO 9935782	A	15-07-1999	AU	2557399 A	26-07-1999
			CA	2316227 A1	15-07-1999
			EP	1050133 A1	08-11-2000
			WO	9935782 A1	15-07-1999
			US	6304658 B1	16-10-2001
			US	2001002486 A1	31-05-2001
WO 9852319	A	19-11-1998	US	5991415 A	23-11-1999
			AU	7568598 A	08-12-1998
			EP	0986873 A1	22-03-2000
			WO	9852319 A1	19-11-1998

EPO FORM P0463

Pour tout renseignement concernant cette annexe : voir Journal Officiel de l'Office européen des brevets, No.12/82

Claims

1. A method of securely implementing a public-key cryptography algorithm, the public key being composed of an integer n that is a product of two large prime numbers p and q, and of a public exponent e, said algorithm also including a private key, said method consisting in determining a set E comprising a predetermined number of prime numbers e_i that can correspond to the value of the public exponent e, said method being characterized in that it comprises the following steps consisting in:

a) computing a value $\epsilon = \prod_{ei \in E} ei$

such that ϵ/e_i is less than $\Phi(n)$ for any e_i belonging to E, where Φ is the Euler totient function;

b) applying the value ϵ to a predetermined computation involving, as a modular product, only the modular product of ϵ multiplied by said private key of the algorithm;

c) for each e_i, testing whether the result of said predetermined computation is equal to a value ϵ/e_i:

- if so, then attributing the value e_i to e, and storing e with a view to it being used in computations of said cryptography algorithm;

- otherwise, observing that the computations of the cryptography algorithm using the value e cannot be performed.

2. A method according to claim 1, characterized in that the cryptography algorithm is based on an RSA-type algorithm in standard mode.

3. A method according to claim 2, characterized in that the predetermined computation of step b) consists in computing a value C:

$C = \epsilon.d$ modulo $\Phi(n)$, where d is the corresponding private key of the RSA algorithm such that $e.d = 1$ modulo $\Phi(n)$ and Φ is the Euler totient function.

4. A method according to claim 2, characterized in that the predetermined computation of step b) consists in computing a value C:

$C = \epsilon.d$ modulo $\lambda(n)$, where d is the corresponding private key of the RSA algorithm such that $e.d = 1$ modulo $\lambda(n)$, with λ being the Carmichael function.

5. A method according to claim 1, characterized in that the cryptography algorithm is based on an RSA-type algorithm in CRT mode.

6. A method according to claim 5, characterized in that the predetermined computation of step b) consists in computing a value C:

$C = \epsilon.d_p$ modulo $(p-1)$, where d_p is the corresponding private key of the RSA algorithm such that $e.d_p = 1$ modulo $(p-1)$.

7. A method according to claim 5, characterized in that the predetermined computation of step b) consists in computing a value C:

C = $\epsilon.d_q$ modulo (q-1), where d_q is the corresponding private key of the RSA algorithm such that $e.d_q = 1$ modulo (q-1).

8. A method according to claim 5, characterized in that the predetermined computation of step b) consists in computing two values C_1 and C_2 such that:

$C_1 = \epsilon.d_p$ modulo (p-1), where d_p is the corresponding private key of the RSA algorithm such that $e.d_p = 1$ modulo (p-1);

$C_2 = \epsilon.d_q$ modulo (q-1), where d_q is the corresponding private key of the RSA algorithm such that $e.d_q = 1$ modulo (q-1);

and in that the test step c) consists, for each e_i, in testing whether C_1 and/or C_2 is equal to the value ϵ/e_i:

- if so, then attributing the value e_i to e and storing e with a view to it being used in computations of said cryptography algorithm;

- otherwise, observing that the computations of said cryptography algorithm using the value e cannot be performed.

9. A method according to claim 3 or claim 4 and in which a value e_i has been attributed to e, said

method being characterized in that the computations using the value e consist in:

choosing a random integer r;

computing a value d* such that $d^* = d+r.(e.d-1)$; and

implementing a private operation of the algorithm in which a value x is obtained from a value y by applying the relationship $x = y^{d^*}$ modulo n.

10. A method according to any one of claims 2 to 4, and in which a value e_i has been attributed to e, said method being characterized in that it consists, after a private operation of the algorithm, in obtaining a value x from a value y, and in that the computations using the value e consist in checking whether $x^e = y$ modulo n.

11. A method according to any one of claims 5 to 8, and in which a value e_i has been attributed to e, characterized in that it consists, after a private operation of the algorithm, in obtaining a value x from a value y, and in that the computations using the value e consist in checking firstly whether $x^e = y$ modulo p and secondly whether $x^e = y$ modulo q.

12. A method according to any preceding claim, characterized in that the set E comprises at least the following e_i values: 3, 17, $2^{16}+1$.

13. An electronic component characterized in that it comprises means for implementing the method according to any preceding claim.

14. A smart card including an electronic component according to claim 13.

15. A method of securely implementing a public-key cryptography algorithm, the public key being composed of an integer n that is a product of two large prime numbers p and q, and of a public exponent e, said method consisting in determining a set E comprising a predetermined number of prime numbers e_i that can correspond to the value of the public exponent e, said method being characterized in that it comprises the following steps consisting in:

a) choosing a value e_i from the values of the set E;

b) if $\delta(p) = \delta(q)$, where δn, $\delta(p)$, and $\delta(q)$ are functions giving the number of bits encoding respectively the number n, the number p, and the number q, testing whether the chosen e_i value satisfies the relationship:

$$(1-e_i.d) \text{ modulo } n < e_i.2^{(\delta(n)/2)+1}$$

or said relationship as simplified:

$$(-e_i.d) \text{ modulo } n < e_i.2^{(\delta(n)/2)+1}$$

where $\delta(p)$, $\delta(q)$, and $\delta(n)$ are the functions giving the numbers of bits respectively encoding the number p, the number q, and the number n;

c) if the test relationship applied in the preceding step is satisfied and so $e = e_i$, storing e with a view to using it in computations of said cryptography algorithm;

- otherwise, reiterating the preceding steps while choosing another value for e_i from the set E until an e_i value can be attributed to e and, if no e_i value can be attributed to e, then observing that the computations of said cryptography algorithm using the value of e cannot be performed.

16. A method of securely implementing a public-key cryptography algorithm according to claim 15, characterized in that it consists in performing step b in the following manner when $\delta(p)\#\delta(q)$, i.e. when p and q are unbalanced, testing whether the chosen e_i value satisfies the following relationship:

$(1-e_i.d)$ modulo $n < e_i.2^{g+1}$

or said relationship as simplified:

$(-e_i.d)$ modulo $n < e_i.2^{g+1}$

with $g=\max(\delta(p), \delta(q))$, if $\delta(p)$ and $\delta(q)$ are known, or, otherwise, with $g=\delta(n)/2+t$, where t designates the imbalance factor or a limit on that factor

17. A method according to claim 15 or claim 16, characterized in that, for all values of i, $e_i \leq 2^{16}+1$, and in that the step b) is replaced by another test step consisting in:

b) if $\delta(p)=\delta(q)$, testing whether the chosen e_i value satisfies the relationship:

$(1-e_i.d) \text{ modulo } n < e_i.2^{(\delta(n)/2)+17}$

or said relationship as simplified:

$(-e_i.d) \text{ modulo } n < e_i.2^{(\delta(n)/2)+17}$

where $\delta(p)$, $\delta(q)$, and $\delta(n)$ are the functions giving the numbers of bits respectively encoding the number p, the number q, and the number n;

otherwise, when p and q are unbalanced, testing whether the chosen e_i value satisfies the following relationship:

$(1-e_i.d) \text{ modulo } n < e_i.2^{g+17}$

or said relationship as simplified:

$(-e_i.d) \text{ modulo } n < e_i.2^{g+17}$

with $g=\max(\delta(p),\delta(q))$, if $\delta(p)$ and $\delta(q)$ are known, or, otherwise, with $g=\delta(n)/2+t$, where t designates the imbalance factor or a limit on that factor.

18. A method according to claim 15 or claim 16, characterized in that step b) is replaced with another test step consisting in:

testing whether the chosen e_i value satisfies the relationship whereby:

the first most significant bits of $(1-e_i.d)$ modulo n are zero;

or said relationship as simplified whereby:

the first most significant bits of $(-e_i.d)$ modulo n are zero.

19. A method according to claim 18, characterized in that the test is performed on the first 128 most significant bits.

20. A method according to any one of claims 15 to 19, characterized in that the cryptography algorithm is based on an RSA-type algorithm in standard mode.

21. A method according to any one of claims 15 to 20, and in which an e_i value has been attributed to e, said method being characterized in that the computations using the value e consist in:

- choosing a random integer r;
- computing a value d* such that $d^* = d+r.(e.d-1)$;

implementing a private operation of the algorithm in which a value x is obtained from a value y by applying the relationship $x = y^{d^*}$ modulo n.

22. A method according to any one of claims 15 to 20 and in which an e_i value has been attributed to e, said method being characterized in that it consists, after a private operation of the algorithm, in obtaining a value x from a value y and in that the computations using the value e consist in checking whether $x_e = y$ modulo n.

23. A method according to any one of claims 15 to 22, characterized in that the set E comprises at least the following e_i values: 3, 17, $2^{16}+1$.

24. A method according to claim 23, characterized in that the preferred choice of the values e_i from the values of the set E is made in the following order: $2^{16}+1$, 3, 17.

25. An electronic component characterized in that it comprises means for implementing the method according to any one of claims 15 to 24.

26. A smart card including an electronic component according to claim 25.